Automated Generation of Tracking Plans for a Network of Communications Antennas

S. Chien, A. Govindjee, T. Estlin[1], X. Wang[2], and R. Hill Jr.[3]
Jet Propulsion Laboratory
4800 Oak Grove Drive, M/S 525-3660
Pasadena, CA 91109-8099
{steve.chien, anita.govindjee}@jpl.nasa.gov

Abstract- This paper describes the Deep Space Network Antenna Operations Planner (DPLAN): a system for automatically generating antenna tracking plans for an automated set of highly sensitive radio science and telecommunications antennas. DPLAN accepts current equipment configuration information and a set of requested track services and uses a knowledge base of antenna operations procedures to produce a plan of activities to provide the services using the allocated equipment. DPLAN produces this plan using and integration of artificial intelligence (AI) techniques of hierarchical task network (HTN) and operator-based planning. We describe the antenna automation problem, the DPLAN system for automatic generation of track plans, current deployment status, and future work.

TABLE OF CONTENTS

1. INTRODUCTION

The Deep Space Network (DSN) [6] was established in 1958 and since then it has evolved into the largest and most sensitive scientific telecommunications and radio navigation network in the world. The purpose of the DSN is to support unpiloted interplanetary spacecraft missions and support radio and radar astronomy observations in the exploration of the solar system and the universe. There are three deep space communications complexes, located in Canberra, Australia, Madrid, Spain, and Goldstone, California. Each DSN complex operates four deep space stations -- one 70-meter antenna, two 34-meter antennas, and one 26-meter antenna. The functions of the DSN are to receive telemetry signals from spacecraft, transmit commands that control the spacecraft operating modes, generate the radio navigation data used to locate and guide the spacecraft to its destination, and acquire flight radio science, radio and radar astronomy, very long baseline interferometry (VLBI), and geodynamics measurements.

From its inception the DSN has been driven by the need to create increasingly more sensitive telecommunications devices and better techniques for navigation. The operation of the DSN communications complexes require a high level of manual interaction with the devices in the communications link with the spacecraft. In

This work was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

[1] Current address: Department of Computer Science, University of Texas, Austin, TX, estlin@cs.utexas.edu
[2] Current address: Rockwell Science Center, Palo Alto, CA, mei@rpal.rockwell.com
[3] Current address: USC/ISI, Marina Del Rey, CA, hill@isi.edu

In more recent times NASA has added some new requirements to the development of the DSN: (1) reduce the cost of operating the DSN, (2) improve the operability, reliability, and maintainability of the DSN, and (3) prepare for a new era of space exploration with the New Millennium program: support small, intelligent spacecraft requiring very few mission operations personnel.

This paper describes Deep Space Network Antenna Operations Planner (DPLAN) which automatically generates plans for individual tracks based on the requested services and equipment allocation. The DPLAN system is one element of a far-reaching effort to upgrade and automate operations of the DSN in order to achieve the three NASA goals mentioned in the last paragraph. We successfully demonstrated a prototype of the DPLAN system in February 1995 at NASA's experimental DSN station, DSS-13 [12,13], on a series of Voyager tracks and efforts are currently underway to insert the technologies used in this demonstration into the operational DSN.

This paper is organized in the following manner. We begin by providing an overview of how the DSN operates. Next we describe an architecture for automating DSN operations -- we give a functional description of each of the components, which includes the Demand Access Network Scheduler (DANS) system for automated resource allocation [5], DPLAN [8][14], an automated procedure generation system, and NMC, a plan execution and monitoring system. In addition we provide examples of the inputs and outputs to each of the components to illustrate what occurs at each step in the process of DSN operations. Next, we describe the DPLAN system; describing: (1) the track plan generation problem; (2) artificial intelligence hierarchical task network (HTN) and operator-based planing; (3) the DPLAN system; and (4) an example of operation. Finally, we describe current efforts to deploy the DPLAN system in the operational DSN and areas of current work.

2. HOW THE DSN OPERATES

Voyager I is cruising at 17.5 kilometers per second toward the outer edge of the solar system. Though its onboard systems are mostly asleep during this phase of its mission, Voyager's health metrics are continually sent to Earth via a telemetry signal radiated by its 40 watt transmitter. It will take eight hours at the speed of light for the signal to reach its destination, Earth, a billion miles away. Upon arrival, the telemetry signal is received by an extremely sensitive ground communications system, NASA's Deep Space Network (DSN), where it is recorded, processed, and sent to the Mission Operations and Voyager project engineers, who assess the health of the spacecraft based on the contents of the signal.

The type of activity just described occurs daily for dozens of different NASA spacecraft and projects that use the DSN to capture spacecraft data. Though the process of sending signals from a spacecraft to Earth is conceptually simple, in reality there are many earthside challenges that must be addressed before a spacecraft's signal is acquired and transformed into useful information.

Network Preparation at the Network Operations Control Center

The first step in a DSN track is called Network Preparation and it occurs at a central control center for the DSN located at JPL, called the Network Operations Control Center (NOCC). The project initiates Network Preparation by sending a request for the DSN to track a spacecraft involving specific tracking services. The DSN responds to the request by attempting to schedule the resources (*i. e.,* an antenna and other shared equipment) needed for the track.

Along with this request, the project prepares a Sequence of Events (SOE) describing the time-ordered activities that should occur during the track. The SOE includes actions that the DSN should take, (*e.g.,* begin tracking the project's spacecraft at 1200 hours), and it also includes events that will occur on the spacecraft being tracked (*e.g.,* the spacecraft will change frequency or mode at a designated time). These events are important because they affect how the DSN provides the services. The project SOE is sent to the DSN, which then generates its own version, called a Ground Network SOE. The Ground Network SOE is a more elaborate version of the project SOE in that it expands the activities from high level descriptions (*e.g.*, begin tracking the spacecraft) into a finer level of detail for use by the operations personnel at the deep space station. The Ground Network SOE is sent to the Deep Space Station (DSS), where the antennas used to perform the actual track are located. Along with the Ground Network SOE a wide range of required support data are transmitted - such as the predicted location of the spacecraft, etc.

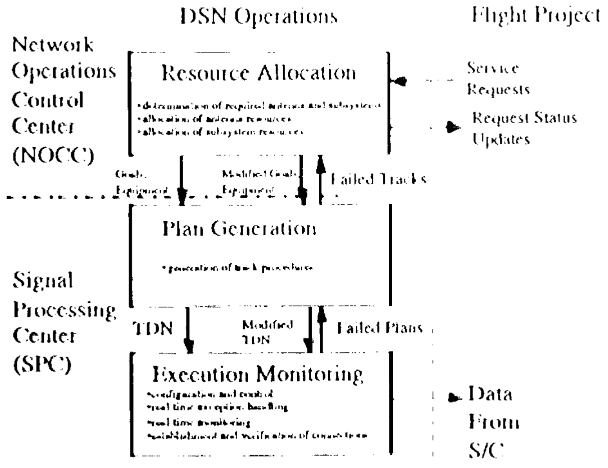

Figure 1: An Automation Oriented View of Deep Space Network Operations

Data Capture at the Signal Processing Center

The data capture process is performed by operations personnel at the deep space station - they determine the correct steps to perform to configure the equipment for the track, perform the actual establishment of the communications link, which we hereafter refer to as a 'link', and then perform the track by issuing control commands to the various subsystems comprising the link. Throughout the track the operators continually monitor the status of the link and handle exceptions (*e.g.*, the receiver breaks lock with the spacecraft) as they occur. All of these actions are currently performed by human operators, who manually issue tens or hundreds commands via a computer keyboard to the link subsystems. The monitoring activities require the operator to track the state of each of the subsystems in the link (usually three to five subsystems), where each subsystem has many different state variables that change over time.

Automation of DSN Processes and the DSN Planner

In the last section we described the current process for transforming a flight project service request into an executable set of DSN operations. As we have already pointed out, many of the steps of the described processes are intensely manual. As part of technology development, demonstration, and deployment, a series of systems are being built to automate various portions of these tasks. The Demand Access Network Scheduler (DANS) [5] is being developed and demonstrated to automate the network resource scheduling process. DANS is designed to work in close coordination with the Network Planning and Preparation (NPP) system which tracks resource usage but does not automatically schedule or reschedule. The DSN Antenna Operations Planner (DPLAN) [3] (see also [2]) is being deployed and further enhanced to automatically generating DSN track plans. The Network Monitor and Control (NMC) system is being deployed to automate the connection operations: intelligent task control, execution monitoring, and exception handling at the DSS sites.

3. 'RACK PLAN GENERA'.'ION: [illegible] PROBLEM

Each day, at sites around the world, NASA's Deep Space Network (DSN) antennas and subsystems are used to perform scores of tracks to support earth orbiting and deep space missions. Because of the complexity of this equipment, the large set of communications services (in the tens), and the large number of supported equipment configurations (in the hundreds, correctly and efficiently operating this equipment to fulfill tracking goals is a daunting task. An additional requirement is that the antenna operations knowledge embodied in the system be easily understandable and maintainable as equipment upgrades, services, protocols, and software changes evolve.

The Deep Space Network Antenna Operations Planner (DPLAN) is an automated planning system developed by the Jet Propulsion Laboratory (JPL) to automatically generate antenna tracking plans to satisfy DSN service requests. In order to generate these antenna operations plans, DPLAN uses a number of information sources, including: the project generated service request, the spacecraft sequence of events, the track equipment allocation, and an antenna operations knowledge base. The service request represents the basic communications services requested during the track (telemetry/downlink, commanding/uplink, ranging (uplink and downlink), etc.).

The sequence of events indicates the relevant spacecraft mode changes (such as transmission bit rate changes, modulation index changes, etc.). The equipment allocation dictates the antenna and subsystem configuration available for the track. The antenna operations knowledge base provides information on the requirements of antenna operations actions; in particular, this information dictates how these actions can be combined to provide essential communications services.

Generation of Tracking Plans - The Inputs and Outputs

The automated track procedure generation problem involves taking a general service request (such as telemetry - downlink of data from a spacecraft) and an actual equipment assignment (describing the type of antenna, receiver, telemetry processor, and so on), and generating the appropriate partially ordered sequence of commands (called a Temporal Dependency Network or TDN; see Figure 3) for creating a communications link to enable the appropriate interaction with the spacecraft. The DSN Antenna Operations Planner (DPLAN) uses an integration of AI Hierarchical Task Network (HTN) and partial order operator-based planning techniques to represent DSN antenna operations knowledge and to antenna operations procedures on demand from the service request and equipment assignment.

The DPLAN planner uses high level track information to determine appropriate steps, ordering constraints on these steps, parameters of these steps to achieve the high level track goals given the equipment allocation. In generating the TDN, the planner uses information from several sources (see Figure 2):

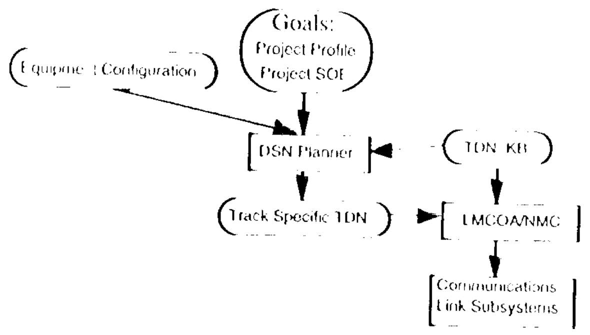

Figure 2: DPLAN and LMCOA/NMC Inputs and Outputs

Project SOE - The project sequence of events specifies events from the mission/project perspective. As a result, the project SOE contains a great deal of information regarding the spacecraft state which is relevant to the DSN track, as well as a large amount of spacecraft information unrelated to DSN operations. Relevant information specified in the project SOE includes such items as the one-way light time (OWLT) to the spacecraft, notifications of the beginning and ending times of tracks, spacecraft data transmission bit rate changes, modulation index changes, and carrier and subcarrier frequency changes.

Project profile - This file specifies project specific information regarding frequencies and pass types. For example, the Project SOE might specify frequency = HIGH, and the project profile would specify the exact frequency used. The project profile might also other signal parameters and default track types.

TDN KB The Temporal Dependency Network (TDN) knowledge base [9][10] stores information on the TDN blocks available for the DSN Planner and LMCOA to use. This knowledge base includes information regarding preconditions, postconditions, directives, and other aspects of the TDN blocks. It also includes information on how to expand the block parameters and name into the actual flatfile entry in a TDN.

Equipment Configuration - This details the types of equipment available and unique identifiers to be used to specify the exact pieces of equipment to be used in the track. These include the antenna, antenna controller, the receiver, and so on.

4. ARTIFICIAL INTELLIGENCE PLANNING TECHNIQUES

AI planning researchers have developed numerous approaches to the task of correct and efficient planning. Two main approaches to generative planning are *operator-based* planners and *hierarchical task network* (HTN) planners. DPLAN uses a combination of both these approach's, exploiting the advantages of each.

Both HTN and operator-based planners typically construct plans by searching in a plan-space. However, they differ considerably in how they search. HTN planners specify plan modifications in terms of flexible, hierarchical and modular task reduction rules, working in a forward-chaining, top down fashion. In contrast, operator-based planners work in a backward-chaining manner, taking a given goal and attempting to resolve its preconditions. Operator-based planners perform all reasoning at the lowest level of abstraction and provide a strict semantics for defining operator definitions.

An HTN planner [7] uses task reduction rules to decompose abstract goals into lower level tasks. HTN planners can encode many types of information into task reductions. By defining or not defining certain reduction refinements, the designer can direct the planner towards particular search paths in certain cases. The user can also directly influence the planner by explicitly adding an ordering constraint or goal protection that would not strictly be derived from goal interaction analyses. Search-control knowledge can also be encoded by writing explicit action sequences to achieve goals, thereby avoiding considerable search.

In contrast, an operator-based planner [15][1] reasons at a single level of abstraction -- the lowest level. Actions are strictly defined in terms of preconditions and effects. Plans are produced through subgoaling and goal interaction analyses. In this framework, all plan constraints (protections, ordering, codesignation) are a direct consequence of goal achievements and action precondition and effect analysis. Thus, an operator-based planner generally has a strict semantics grounded in explicit state representation, *i.e.* defining what is and is not true in a particular state (or partial state).

DPLAN combines both these two methods, exploiting the advantages of each. An operator-based planner requires a rigid representation which is both a strength and a weakness. It is advantageous since it more explicitly directs the knowledge engineer in encoding a domain. Yet, it can also make certain aspects of a domain difficult to represent. Known ordering constraints and operator sequences can be difficult to encode if they cannot easily be represented in terms of preconditions and effects. Such constraints can and are often forced by adding "dummy" preconditions, however this solution can often create a misleading representation. An HTN planner, on the other hand, allows, the easy representation of known ordering constraints. Domain information is easily represented within the HTN framework allowing the capability of directing search through explicitly defined ordering constraints and goal protections.

Using a combination of both HTN planning and operator-based planning allows DPLAN to direct search and define knowledge in a top down fashion, but also to define knowledge in the more structured operator based fashion without the problem of raving to create "dummy" variables.

In the integrated HTN/operator framework that DPLAN utilizes, the planner can use the two planning methods and reason about different types of planning goals. In DPLAN, we have defined two main goal types: *activity-goals* and *state-goals*. *Activity-goals* correspond to operational or non-operational domain activities and are manipulated using HTN planning techniques. Operational activity-goals are considered primitive tasks that can be directly executed. Non-operational activity-goals must be further decomposed into operational ones through HTN reduction rules. *State-goals* are defined as the preconditions and effects of activity-goals, and are achieved through standard operator-based planning methods.

DPLAN uses both hierarchical task network (HTN) planning techniques and operator-based planning techniques. In HTN planning, abstract actions such as "calibrate receiver" or "configure sequential ranging assembly" are decomposed into specific directives for specific hardware types. In operator-based planning, requirements of specific actions are satisfied using means-end analysis, which matches action preconditions to effects and resolves any occurring ordering constraints.

5. THE DPLAN PLANNING ALGORITHM

The DPLAN planning algorithm uses a unique combination of the HTN and operator-based planning techniques discussed above. DPLAN operates by refining a set of input top-level goals into a set of low-level operational goals. Plans are represented by a three-tuple: <U,C,S> where U is a set of non-operational (or high-level) goals, C is a set of constraints, and S is a set of operational-goals. At the end of planning, U should be empty and the goals in S are returned as the final plan steps.

A overview of DPLAN algorithm is shown in Figure 3. The main inputs to DPLAN are: a set of high-level goals G, a set of decomposition rules R, and the set of all possible operational goals O. Search is implemented by keeping a queue of partial plans to be explored. Currently, plans are selected from the queue using a best-first heuristic, however, other search techniques could easily be employed. Step 1 of the main loop selects the best plan off the queue, and Step 2 checks if that plan is a solution. If no solution has been found then a new goal is selected for refinement. Step 5 chooses a refinement strategy for that goal, and in Step 6, any new plans created through that strategy are inserted into the plan queue. A plan is considered a solution if two conditions are true.

```
Algorithm DPLAN (G,R,O)

Initialize the plan queue Q := (<G,{}, {}>)
While Q is not empty and the resource bound
has not been exceeded,

1. Select a promising plan P in Q using
   heuristics,
2. Remove P from Q
3. If P contains only operational-goals, then
   check context goals in P. If the context
   goals are achieved, return P.
   Otherwise goto 1.
4. Choose a non-operational goal g from U
5. Refine g.
6. Insert any new plans generated by
   refinement into Q.
```

Figure 3 - The DPLAN Search Algorithm

The first is that there are no non-operational goals left to be refined. The second condition is that all context goals have been achieved or are directly achievable in the current plan. Context goals are goals which were needed for applying a decomposition rule, but are supposed to be accomplished by some other part of the plan. If all context goals have been achieved, then the plan is returned as a success.

DPLAN can use several different refinement strategies to handle non-operational goals. There are two main types of goals in DPLAN: activity-goals and state-goals. Activity-goals correspond to operational or non-operational activities and are usually manipulated using HTN planning techniques.

State-goals correspond to the preconditions and effects of activity-goals, and are achieved through operator-based planning. State-goals that have not yet been achieved are also considered non-operational. Figure 4 shows the procedures used for refining these two types of goals. As soon as a refinement strategy is applied to an activity-goal or state-goal, it is removed from the list of non-operational goals.

If *g* is an Activity-Goal,

1. Decompose: For each decomposition rule r in R which can decompose *g*, apply r to produce a new plan P', If all constraints in P' are consistent, then add P' to Q.
2. Simple Establishment: For each activity-goal *g'* in U that can be unified with *g* simple establish *g* using *g'* and produce a new plan P'. If all constraints in P' are consistent, then add P' to Q.

If *g* is a State-Goal,

1. Step Addition: 'o each activity-goal effect that can unify with *g*, add that goal to P to produce a new plan P'. If the constraints in P' are consistent, then add P' to Q.
2. Simple Establishment: For each activity-goal *g'* in U that has an effect e that can be unified with *g*, simple establish *g* using e and produce a new plan P'. If all constraints in P' are consistent, then add P' to Q.

Figure 4 - Goal Refinement Strategies

DPLAN can also use additional domain information for more efficient and flexible planning. For instance, a planning problem can specify a list of static context facts. These facts represent operational goals that are always considered to be true. Such goals are easy for DPLAN to verify during planning and can help in pruning off search branches. Other possible inputs include sets of preconditions and effects for operational activities, a set of final goals that must be true in the plan solution, and a set of initial goals that are true at the beginning of planning. This information is not required for standard DPLAN operation, but can be very beneficial during planning.

An Example of DPLAN representation

As mentioned in the preceding section, DPLAN uses several different types of knowledge to construct a plan. A main component of this knowledge is a set of decomposition rules. These rules specify how the planner can break down nonoperational activity-goals into lower-level operational goals. A sample rule for performing a telemetry antenna track is shown in Figure 5. This rule defines how the general telemetry operation is broken down into steps. The left-hand side (LHS) of a decomposition rule consists of a set of initial goals, and possibly, a number of other constraints that specify when the rule should be applied. All initial goals and specified constraints must be true in the current plan for the rule to be selected. The initial goals of a rule are the nonoperational goals that the rule "decomposes" into lower-level goals. The rule shown above has only one initial goal that checks if a *telemetry track-goal* is present in the current plan.[4]

The right-hand side (RHS) of a rule contains a set of new goals and constraints over those goals. Once a rule is applied, these new goals replace the LHS initial goals in the current plan. The RHS also contains ordering constraints and protections that specify information about the new goals. An ordering constraint specifies that two goals must be placed in a certain partial order in the final TDN. A protection specifies a causal link that exists between goals, where a causal link explains how the effect of one goal achieves the precondition of another goal. This link must always be preserved in order to generate a correct plan. Ordering constraints and protections are added to the current plan and must always be kept consistent during planning. For instance, if an ordering constraint is violated during planning, then the current plan is discarded,

[4] Other possible LHS constraints include additional goal conditions that must be present in the plan, context goals, which the planner expect to be achieved by another rule, and codesignation constraints, which check whether two variables can or cannot be unified.

```
(decomprule default telemetry-track
  lhs
    (initialgoals ((track-goal spacecraft-track telemetry ?track-id)))
  rhs
    (newgoals
        ((g1 (perform-antenna-controller- configuration ?track-id))
         (g2 (configure-metric-data-assembly ?track-id))
         (g3 (perform-microwave-controller-configuration ?track-id))
         (g4 (perform-receiver-configuration ?track- id))
         (g5 (perform-telemetry-configuration ?track- id))
         (g6 (move-antenna-to -point ?track-id))
         (g7 (perform-receiver-calibration ?track- id)))
     constraints
         ((before g1 g6)
          (before g7 g3)
          (before g4 g7))))
```

Figure 5 - Decomposition rule for telemetry track

```
(decomprule configure-receiver1
  lhs
    (initialgoals ((perform-receiver-configuration ?track- id)))
      conditions ((CCN-equipment-assignment ?track-id ?equip)
                  (isa ?equip BLOCK-IV-RECEIVER)))
  rhs
    (newgoals ((configure-block-iv-receiver ?track- id ?equip))))


(decomprule configure-receiver2
  lhs
    (initialgoals ((perform-receiver-configuration ?track-id)))
     conditions ((CCN-equipment-assignment ?track - id ?equip)
                 (isa ?equip BLOCK-V-RECEIVER)))
  rhs
    (newgoals ((configure-block-v-receiver ?track-id ?equip))))
```

Figure 6 - Two rules for decomposing the *perform-receiver-configuration* goal

```
(calibrate-transmitter
      :parameters (?track-id)
      :preconditions ((exciter-configured ?track- id)
                      (microwave- controller-configured ?track-id)
                      (transmitter-configured ?track-id))
      :effects (((transmitter-calibrated ?track-id))))
```

Figure 7- Schema for *calibrate-transmitter* goal

and the planner selects another plan from the queue. to work on.

Sometimes there may be several different rules that can be used to decompose the same initial goal. For instance, in tracks for 70m antennas, there are several different methods for configuring a receiver depending on the type of receiver being used. To represent these different methods, there are several different rules that can be used to decompose the *perform-receive-configuration*, which was asserted by the telemetry rule in Figure 5. The rules listed in Figure 6 show two possible ways to break down this goal. The first rule states that if the current goal is to configure the receiver, and the receiver assigned to the antenna track is a Block-IV receiver, then the configuration method for Block-IV receivers should be used. The second rule states a similar method for Block-V receivers. Thus, as shown in these examples, decomposition rules can be used to represent both specific and general domain knowledge.

Another type of knowledge used by DPLAN is a set of activity-goal schemas. These schemas define the parameters and the preconditions and effects that are associated with an activity-goal. As explained in Section 5, activity-goal preconditions and effects correspond to *state-goals* and are manipulated through operator-based planning techniques. A sample of an activity-goal schema is shown in Figure 8. This schema definition shows the associated parameters and the preconditions and effects of the *calibrate-transmitter* task. For instance, this schema reflects that it is necessary to configure the exciter before calibrating the transmitter. Since DPLAN employs a combination of operated-based and HTN planning techniques, a variety of knowledge types can be exploited by the planner. These different knowledge formats allow domain knowledge to be more naturally represented than if only one format were utilized. Each format allows for a different type of knowledge encoding. For instance, decomposition rules allow for the representation of abstract levels of domain objects and goals. Allowing abstract representations of these items, allows the user to represent domain information in a more object-oriented form, which is easier to write and reason about. This format also contributes to a more general domain knowledge base that can be efficiently updated and maintained.

Conversely, the utilization of goal schemas and operator-based planning techniques allows certain constraint information to be more easily expressed in the domain. For instance, ordering constraints that are due to precondition-effect interactions are directly deduced during planning, instead of having to be explicitly listed by the user. In particular, ordering constraints that apply to very specialized goals, as opposed to very general ones, can be more easily expressed through precondition/effect schemas, than through decomposition rules. For more information on the advantages and disadvantages of employing HTN and operator-based planning techniques for this type of domain see [4].

An Operations Example

In order to begin the planning process, DPLAN is provided with a problem specification that contains several lists of information. Specifically, each problem contains a list of decomposition goals, along with possible lists of initial state predicates, static state predicates, and final state predicates. A sample problem for performing telemetry and ranging with a 70m antenna is shown in Figure 8.

The *init-state* field specifies a list of propositions that are true in the initial state of the planner. For instance, as shown in the above problem, the exciter drive is assumed to be off prior to when the track is performed. The *static-state* field specifies a list of propositions that are always true during planning *(i. e.,* can never be deleted), and is commonly used to list equipment types available to the track. The *decompgoals* field holds the list of nonoperational goals that are to be broken down into lower-level goals through the use of decomposition rules. The *final-state* field is a list of propositions that must be true in the final plan. The *init-state*, *static-state* and *final-state* fields are not necessary for standard planner operation, and can be left empty. However, these fields are very beneficial for increasing planner efficiency by providing extra domain knowledge. Other inputs to the planner, include a list of decomposition rules and a list of goal schemas, which were explained in the previous section.

```
(decompproblem TELEM70
  (init-state((exciter-drive-off track1)
              (range-mode-off track1)
              (test-translator-off track1)))
  (static-state
    ((CCN-equipment assignment track1 bstring1)
     (isa bstring1 type-B-telemetry-string)
     (CCN-equipment assignment track1
                                   AI'A-7OIII)
     (isa AI'A-7OIII AI'A)
     (CCN-equipment-assignment track1 bvr1)
     (isa bvr1 BVR)
     (CCN-equipment assignment track1 rec1)
     (isa rec1 1{1.(')
     ((''N equipment assignment track1 ugc1)
     (isa ugc1 UGC)))
  (decompgoals
    ((perform-pre-cal track1)
     (track-goal spacecraft-track telemetry track1)
     (track-goal spacecraft track ranging track1 )))
  (final-state () ))
```

Figure 8 - Problem specification for a telemetry and ranging track

DPLAN is currently started by executing the following command from the UNIX prompt:[5]

```
dplan  <problem-string>     <output-
filename> <annotation-filename>
```

The problem-string input is a problem name (*e.g.* 34m, 70m). When this string is given, the planner will expect to find the following files:

```
rules-<problem-string>
goals-<problem-string>
prob-<problem-string>
```

The "rules" files specifies the list of decomposition rules, the "goals" file specifies the list of goal schemas and the "prob" file specifies the particular problem specification (including initial state, decomp goals, etc.). DPLAN will parse the information in these files (using GNU tools flex and bison) into a usable form. Then, using the algorithm introduced in Section 5, DPLAN will generate a plan that successfully achieves all *decompgoals* and any *final-state* goals listed in the problem specification.

A final plan contains a large amount of information, including a list of operational goal names (corresponding to TDN blocks), a list of ordering constraints over those goals, and a list of annotations that describes how the plan was built (*i. e.*, what rules and operations were use.(l). Currently the planner outputs this information in the following way. Three output files are created, a text output file, an annotation file, and a graph-input file. The text output file contains a textual listing of blocks and parameters where blocks are listed in a correct ordering (*i. e.*, blocks *do* no violate any plan ordering constraints). The annotation file contains a textual list of annotations describing the plan. The graph-input file contains a list of node names and ordering constraints, which can be used to construct a graphical representation of the plan. See Figure 9 for an example of a plan generated (a TDN) for a problem specification such as that shown in Figure 9.

6. CURRENT STATUS

The knowledge base for the planner currently supports all the antenna types at the DSN[6]. All valid types of spacecraft passes for each antenna type are implemented in the knowledge base.

Spacecraft passes include the following:

- *telemetry* Telemetry is a downlink with the spacecraft where information is relayed from the spacecraft to the DSN station on earth.

- *ranging* Ranging is a method of finding the distance between the spacecraft and the earth which requires both an uplink and a downlink to the spacecraft.

[5] Future development plans include building a more sophisticated user interface that will allow the user to easily interact with the planner.

[6] Excepting the 26-meter antenna which may be phased-out in the near future.

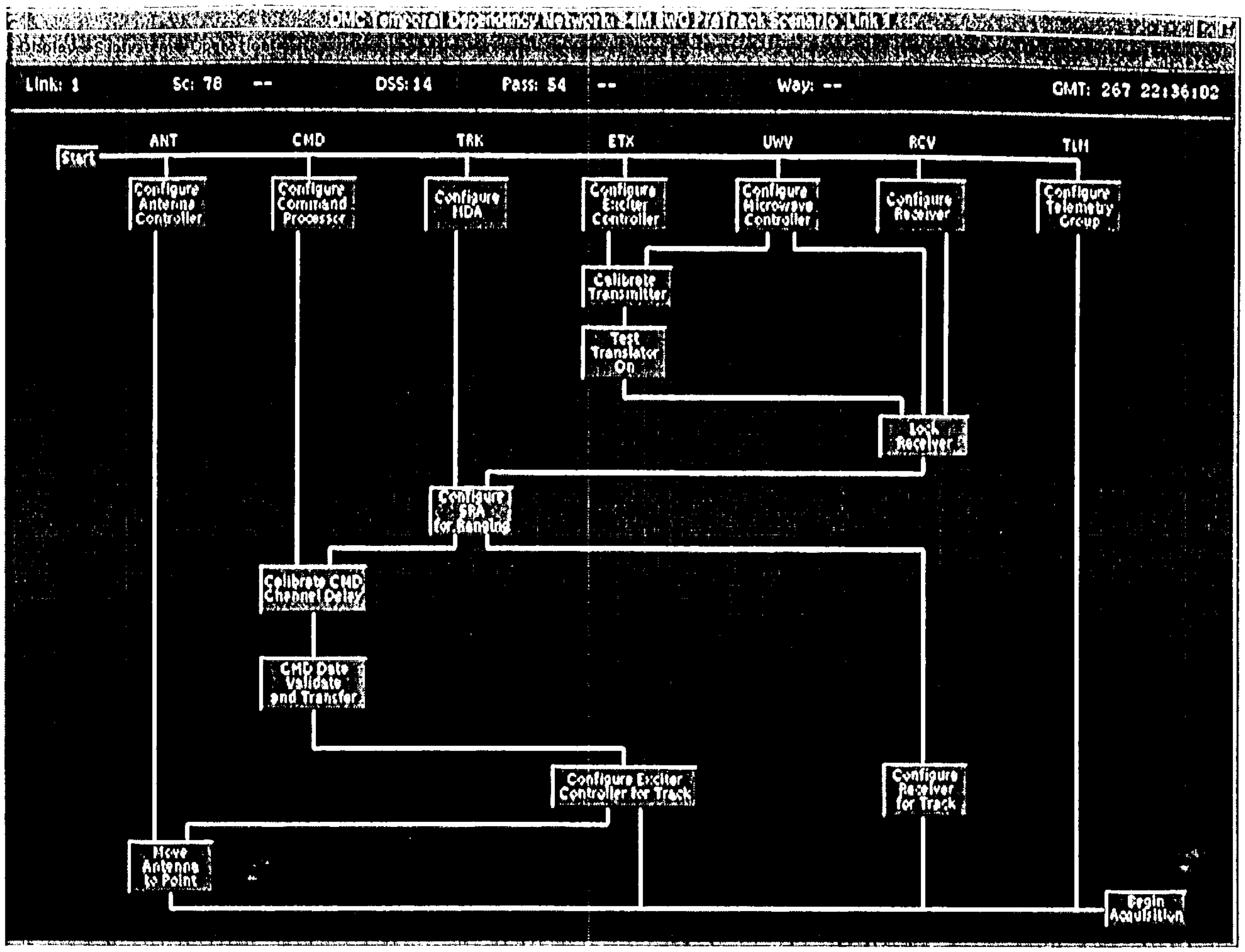
OMC Temporal Dependency Network: 34M BWG PreTrack Scenario Link 1
Link: 1
Sc: 78
--
DSS: 14
Pass: 54
--
Way: --
GMT: 267 22:36:02
ANT
CMD
TRK
ETX
UWV
RCV
TLM
Start
Configure Antenna Controller
Configure Command Processor
Configure MDA
Configure Exciter Controller
Configure Microwave Controller
Configure Receiver
Configure Telemetry Group
Calibrate Transmitter
Test Translator On
Lock Receiver
Configure SRA for Ranging
Calibrate CMD Channel Delay
CMD Data Validate and Transfer
Configure Exciter Controller for Track
Configure Receiver for Track
Move Antenna to Point
Begin Acquisition

Figure 9 - Temporal Dependency Network for 34 M Beam Wave Guide Antenna Pre Track for Telemetry, Commanding, and Ranging Services

- *commanding* Commanding is an uplink to the spacecraft where commands are sent from the DSN station to the spacecraft in order to cause the spacecraft to carry out given tasks.

- *VLBI ΔDOR* VLBI (Very Long Baseline Interferometry) uses quasars distant space objects in order to perform certain operations. VLBI ΔDOR provides information on the spacecraft's angular position by doing simultaneous observations from two stations of the spacecraft, a quasar, and then an observation of the spacecraft again in order to gather doppler data. This data is then used to determine how to maneuver the spacecraft through space to its destination.

- *VLBI clock sync* VLBI clock sync gives the [illegible] position of two stations relative to a quasar in order to determine the rate of change of the clocks at the two DSN stations.

- *radio science* For radio science, the spacecraft downlinks Radio Frequency (RF) signal information that it has collected. The RF signals are used by scientists in [illegible] experiments.

Not all antenna types do all types of spacecraft passes. For example, the 34m STD antenna is not used for any type of

VLBI activity. For each of the antenna types that DPLAN covers in its knowledge base, all the types of spacecraft passes that the antenna is used for, are covered in the DPLAN knowledge base:

- *34m BWG* 34-meter Beam Wave Guide. Telemetry, commanding, and ranging.
- *34m STD* 34-meter Standard. Telemetry, commanding, and ranging.
- *34m HEF* 34-meter High Efficiency. Telemetry, commanding, ranging, VLBI ADOR, and radio science.
- *70m BVR* 70-meter with Block V Receiver. Telemetry, commanding, ranging, VLBI ADOR, VLBI clock sync, and radio science.
- *70m BIVR* 70-meter with Block IV Receiver. Telemetry, commanding, ranging, VLBI ADOR, VLBI clock sync, and radio science.

Generating a plan to make an antenna operational and ready to communicate with a given spacecraft is a complex process. The starting up and turning on of different pieces of equipment and subsystems must be carefully coordinated. DPLAN's knowledge base covers information on the full-range of antenna equipment from the primitive on/off operation of the test translator to the sophisticated operation of the AGC (antenna gain controller) or the UGC (microwave controller). Information on all subsystems necessary to antenna operation is also covered, such as the receiver subsystem, the antenna subsystem, the exciter-transmitter subsystem, and others. DPLAN's knowledge of the functioning of the equipment and subsystems is used to determine in which order the TDN blocks associated with each piece of equipment and each subsystem must be executed.

The total number of rules in the knowledge base (covering all antenna and track types) is 197: 91 decomposition rules (average of 23 decomposition rules per antenna type) and 106 goal schemas. The knowledge base is modular and easily extended to accommodate new antenna types and new subsystems or equipment types. Also, as changes are made to existing antennas, equipment, and subsystems, the rules can easily be modified. For example, if a new type of antenna controller is added to the 34m-HEF antenna, then it is simply a matter of adding in a new rule that configures the new antenna controller. Other rules which use the antenna controller rule do not need to be changed because of the decomposition structure of the knowledge base.

All the plans generated by the planner for the different antenna types and their valid spacecraft passes (including a majority of the multiple combinations of passes) have been verified by the DSN operator experts from all three of the DSN complexes: Goldstone, California (October 1995), Madrid, Spain (January 1996), and Canberra, Australia (May 1996). For example, the 34m-STD antenna can support telemetry, ranging, and commanding spacecraft passes, and any combination of those three types of passes. DPLAN generated the resulting 7 combinations of spacecraft passes (telemetry, ranging, telemetry & ranging, telemetry & commanding, etc.) which were verified on paper by the various operator experts as being correct, executable plans in terms of the ordering of the TDN blocks and the inclusion (or exclusion) of sufficient and necessary TDN blocks.

More testing will occur during the integration phase. During integration, the plans generated by DPLAN are executed by the Automation Engine (AE), by firing scripts associated with each TDN block in the plan. The scripts execute 'operator directives' which turn on and off pieces of equipment, configure subsystems, move the antenna, etc.

A preliminary demonstration was successfully done integrating the planner with the other elements which comprise the DSN automation. The planner successfully made a plan which was then executed (a simulation) by the AE. This took place in December 1995. Further testing of the planner took place in August 1996 in a computer simulated antenna environment with simulated subsystems and equipment. During final integration, anticipated in August 1997, DPLAN will be fully integrated: the AE will call the planner to generate a given plan and then execute that plan, firing off the necessary scripts for the TDN blocks. This will first be tested in the antenna simulator environment and then tested at the Goldstone, California DSN Complex.

7. DISCUSSION

In this section we discuss several issue.s relevant to the DSN planner including: comparison to alternative methods of automation, representation for maintainability, plan quality, and replanning.

Representation for Maintainability

An important aspect of the DPLAN representation is that it allows for natural encoding of abstract objects and procedures (*e.g.*, receiver calibration). By allowing decomposition rules to refer to abstract objects, changes to DSN procedures involve fewer knowledge base updates than if the knowledge base contained a large number of very specific rules. For instance, a change relating to a specific equipment type. need not affect more general domain information. If a new receiver type called a BLOCK-VI receiver were added to the DSN equipment list, more general rules, such as the telemetry rule shown in Figure 6, would not need to be modified. Instead, only a few more specific rules need be constructed or edited. In this case., arlcw(:{\~{fi~(4 rc-rccci\~cl rule would be added to the set of rules shown in Figure 7. Therefore, many such changes would cause only a few specialized rules to be created or updated instead of causing numerous rules to be modified. Even with the current DSN goal k) automate all TDN generation, the planning knowledge base must be constantly updated and verified. Fewer more general rules are cheaper to update and verify, and thus support more efficient knowledge base maintenance.

Another benefit to this type of representation is that domain information is more easily understood. By keeping domain details separate from more general knowledge, it is easier for a user to understand the general aspects of all antenna track. For example, to understand the general steps of a telemetry operation, a user only has to view the main telemetry track decomposition rule. If more low-level knowledge is desired, such as how to operate a particular piece of equipment, the user could then search for rules that directly pertain to that equipment type.

Comparison to Scripts

One option considered by DSN personnel was to implement the higher level of track automation by a hierarchy of scripts. There would be scripts for general activities , such as calibrating a Block V Receiver in the context of a ranging track. This scripting approach can be viewed as similar to the HTN planning approach, but with two key differences. First, the.rc. is no explicit representation of the context in which a script will necessarily achieve the goal. The set of situations in which a script S is expected to work is represented only implicitly in the set of scripts which call the script S. 'I'tic! intended coverage, conditions, etc. are not explicitly represented as the.y are, in HTN rules. The second difference is that the planner allows a "call by goal" usage in operator-based planning. In this way, the planner can invoke routines (operators) by the conditions it desires to achieve, and the planner will automatically detect and rc.solve interactions with other activities. Not only does the planner representation allow for encoding of conditions and assumptions of when particular activities are appropriate (through conditioned on HTN rules 0 1 preconditions on operators), it actually requires such definitions in order (o operate correctly. Therefore it encourages correct documentation of operations requirements of activities.

Comparison to End-to-End TDNs

Another option considered by DSN Operations was to simply encode end-to-end TDNs fOr each of the supported combinations of the C1OSS product of service request combination and equipment allocation. However, this option has several drawbacks. First, articulating all of the relevant knowledge in this format can be very tedious and prone to error. The expert operators, while generating the initial set of end-to end TDNs said that they often found it difficult to keep all of the different TDNs straight. Second, this representation is not amenable to maintenance. If an equipment type is added or changed, it must be changed in every TDN which is relevant. The knowledge pertaining to the equipment type if not centralized in a set of rules or activity definitions as in the planning representation.

Representing and Reasoning about Plan Quality

Representing and reasoning about plan quality [11][16][17][18] is another key concern of DSN operations. Since there is often more than one correct plan for a particular antenna operation, it is important for a planning system to be able to compare a set of final plans using user identified plan quality measures. There are a number of quality measures that can be emphasized during planning including producing more robust, flexible and/or efficient plans. One important quality goal is to minimize the overall plan execution time. In particular, the time to setup (pre-calibration) and reset (post-calibration) the communications link can often be reduced. For instance, it can take up to two hours to manually pre-calibrate a DSN 70 meter antenna communications link for certain types of mission. By using a plan generated by DPLAN, this time can be reduced to approximately 30 minutes, where further reductions in set-up time are limited by physical constraints of the subsystems themselves. Minimizing plan execution time allows more data to be returned per operating time for the link.

Plan execution time can often be significantly reduced by exploiting parallel path possibilities, especially where the control of multiple subsystems is involved. DPLAN currently uses the critical path length of a plan to help identify better plans. Critical path length is calculated using time information attached to a TDN block, which specifies the average time it should take to execute the block. By comparing critical path lengths of competing plans, DPLAN can choose a highly efficient final plan that will provide a minimal execution time.

Another important measure of plan quality is generality. Because of the considerable effort involved in generating, maintaining, and refining TDNs, a single generalized TDN is cheaper than hundreds or thousands of experiment-specific TDNs. For example, one of the missions frequently performed in the DSN domain is called the Ka-band Antenna Performance (KaAP) experiment. The KaAP TDN currently produced is considered a generalized TDN since it represents the many different ways that a KaAP experiment can be executed. The support data for each particular KaAP experiment identifies a particular path through the TDN. This path can change depending on the particular mission schedule. In particular, there is a data capture loop in the KaAP TDN that allows data to be captured from either a star or a planet, thus requiring different antenna modes. One experiment may specify that data be acquired from the following sources in sequence: star1, star2, star3. Whereas another experiment may specify that data by acquired from: star 1, planet1, star1, star2. This more general TDN helps provide for more efficient knowledge maintenance since only one TDN must be maintained for this track.

Flexibility is another aspect of plan quality that has been a requirement in the DSN domain. For instance, the support data for a particular experiment may specify a particular path through the TDN, however, the operator has the flexibility to alter this path in real-time. The final plan must be flexible enough to handle these real-time changes. Some of the changes that the operator can make to the TDN are skipping blocks, deleting commands in blocks, adding commands in blocks, and editing time tags on blocks. It may also be necessary (or desirable) for an operator to reorder blocks. For example, some TDN blocks cannot execute in parallel due to resource conflicts. The ordering of such blocks can often affect plan quality by making a plan more robust or more efficient, depending on the particular antenna operation and current track status. If a better ordering is known prior to TDN generation, this information can be input to the planning system which will incorporate it into the final TDN. However, these ordering constraints may often be best determined at runtime by the operator.

There are also some standard TDN blocks that may be inserted into a plan at various points (such as transmission rate changes, etc.). If such commands are executed in the middle of an inflexible plan, it may not be possible to continue execution. Depending on the steps inserted, preconditions, postconditions, and time tags of other blocks may become invalid. Flexible plans that allow for the insertion of common steps while still retaining their applicability are greatly valued.

A final plan quality issue is robustness. Ideally, the final plan representation should be expressive enough to provide robustness under a variety of situations; however, an expressive representation usually increases an application's complexity and often results in a loss of generality. In the DSN application, the TDN representation used to represent the final plan has initially been kept extremely simple, although it does include parallelism. As the intricacies of particular antenna procedures became evident, more expressive representations may be required. Constructs such as loops, metric time, and actions with temporal scope could be added. Unfortunately, this may cause the creation of very specific constructs that are very specific to a certain track. For instance, in the Ka-band Antenna Performance thick, a useful planning construct to have is a "loop until time" construct, which is used when the actions in a loop need to be executed until a pre-specified time occurs. So far, such a construct has been deemed necessary for only one particular kind of antenna track, and thus, may not be applicable in other tracks. By adding such a construct, the plan representation does become more flexible and may provide for more robust plans. However, such an addition also increases planning complexity. This trade-off may be necessary in order to generate plans that are robust enough to be correctly executed in such a varying and complicated domain.

Replanning for Antenna Tracks

At the level of track procedure generation DPLAN is also required to replan during the course of typical antenna operations. Replanning occurs in two general cases. First, sometimes after a plan has been generated, the objectives may change (exactly the changing objectives item in the taxonomy). Often shortly prior to or during a track a project may submit a request to add services to a track. These correspond to additional goals to be incorporated into the track plan. In the case of goals added before the track actually begins, DPLAN addresses this problem by adding these additional unachieved goals to the current plan and restarting the planning process with this single parent plan. This method is incomplete in theory because the planner may have made choices which are incompatible with the new goals. However, for the specific sets of goals and domain theories related to antenna operations we have examined we have been able to use encodings for which completeness has not been a problem. But this is an area of current work. In the case of goals added during the actual track, we have not addressed this case - it is an area of current work.

Another case for replanning for DPLAN is due to dynamism. After a plan has been generated, a block (plan step) may fail, a piece of equipment may require resetting (due to general unreliability), or a piece of equipment may be fail 01 be pre-empted by a higher priority track. In the case of a simple plan step failure DPLAN simply calls for re-execution of the block. If a piece of equipment requires resetting, DPLAN has knowledge describing which achieved goals are undone and require re-establishment. DPLAN then uses a replanning technique [20] which re-uses parts of the original plan as necessary to re-achieve the undone goals. This technique takes advantage of the fact that the original plan begins from a state which is equivalent to resetting all of the subsystems.

8 CONCLUSIONS

This paper has described the DSN Antenna Operations Planner (DPLAN) which automatically generated communications antenna tracking plans based on requested services and equipment allocation. DPLAN uses a knowledge base of information on tracking activities and a combination of artificial intelligence hierarchical task network (HTN) and operator-based planning methods to generate appropriate tracking plans. We have also described the deployment status of the DPLAN system and outlined areas of current work including representation and reasoning about plan quality, replanning, and representation to support maintainability.

REFERENCES

[1] J.G. Carbonell, J. Blythe, O. Etzioni, Y. Gil, R. Joseph, D. Kahn, C. Knoblock, S. Minton, M. A Perez, S. Reilly, M. Veloso, X. Wang, "Prodigy 4.0: The Manual and Tutorial," Technical Report, School of Computer Science, Carnegie Mellon University.

[2] S. A. Chien, R. W. Hill Jr., X. Wang, T. Estlin, K. V. Fayyad, and H. B. Mortensen, "Why Real-world Planning is Difficult: A Tale of Two Applications," in New *Directions in AI Planning*, M. Ghallab and A. Milani, ed., IOS Press, Washington, DC, 1996, pp. 287-298.

[3] S. Chien, A. Govindjee, X. Wang, T. Estlin, R. hill, Jr., "Integrating Hierarchical Task Network and Operator-based Planning Techniques to Automate Operations of Communications Antennas," *IEEE Expert*, to appear, Winter 1996.

[4] S. Chien, X. Wang, and T. Estlin, "Hierarchical Task Network and Operator-Based Planning: Competing or Complementary?," JPL Technical Document D- 13390, NASA Jet Propulsion Laboratory, Pasadena, CA, January 1996.

[5] S. Chien, R. Lam, and Q. Vu, "Resource Scheduling for a Network of Communications Antennas," Proceedings of the 1997 IEEE Aerospace Conference, Aspen, CO, February 1997.

[6] Deep Space Network, Jet Propulsion Laboratory Publication 400-517, April 1994.

[7] K. Erol, J. Hendler, and D. Nau, "UMCP: A Sound and Complete Procedure for Hierarchical Task Network Planning," *Proceedings of the Second International Conference on AI Planning Systems*, Chicago, IL, June 1994, pp. 249-254.

[8] "T. Estlin, X. Wang, A. Govindjee, and S. Chien, "DPLAN Deep Space Network Antenna Operations Planner Programmers Guide Version 1.0," JPL Technical Document D-13377, Jet Propulsion Laboratory, California Institute of Technology, February 1996.

[9] Fayyad, K.E. and L. P. Cooper. "Representing Operations Procedures Using Temporal Dependency Networks," Space.OIM '92, Pasadena, CA, November 1992.

[10] K. Fayyad, R.W. Hill, Jr., and E.J. Wyatt. "Knowledge Engineering for Temporal Dependency Networks as Operations Procedures." *Proceedings of AIAA Computing in Aerospace 9 Conference*, 1993, San Diego, CA.

[11] J. M. Gratch, S. A, Chien, and G. F. DeJong, "Learning Search Control Knowledge to Improve Schedule Quality," *Proceedings of the 1993 Workshop on Knowledge-based Production Planning, Scheduling, and Control*, Chambery France, August 1993, pp. 159-168.

[12] R. W. Hill, Jr., S. A. Chien, and K. V. Fayyad, "Automating Operations for a Network of Communications Antennas," *Proceedings of the 1996 IASTED International Conference on Artificial Intelligence, Expert Systems, and Neural Networks*, Honolulu, HI, August 1996.

[13] R. W. hill, Jr., S. A. Chien, K. V. Fayyad, C. Smyth, T. Santos, and R. Bevan, "Sequence of Events Driven Automation of the Deep Space Network," *Telecommunications and Data Acquisition* 42-124, October-December 1995.

[14] hill, R. W., Jr., S. Chien, C. Smyth and K. Fayyad, "Planning for Deep Space Network Operations" Proceedings of the 1995 AAAI Spring Symposium on Integrated Planing Applications, Palo Alto, CA, 1995, AAAI Press.

[15] J. S. Penberthy and D. S. Weld, "UCPOP: A Sound Complete, Partial Order Planner for ADL," *Proceedings of the Third International Conference on Knowledge Representation and Reasoning*, October 1992.

[16] M. Williamson and S. Hanks, "Optimal Planning with a Goal-directed Utility Model," Proceedings of the Second International Conference on AI Planning Systems, Chicago, IL, June 1994, pp. 176-180.

[17] M. Williamson and S. Hanks, "Flaw Selection Strategies for Value-directed Planning," Proceedings of the Third International Conference on AI Planning Systems, Edinburgh, UK, May 1996, pp. 237-244.

[18] A. Perez and J. Carbonell, "Control Knowledge to Improve Plan Quality," Proceedings of the Second International Conference on AI Planning Systems, Chicago, IL, June 1994, pp. 323-328.

[19] Final Report of the Services Fulfillment Reengineering Team, JPL Interoffice Memorandum, March 14, 1995.

[20] X. Wang and S. Chien, "Replanning for the Deep Space Network (DSN) Antenna Operations Planner Preliminary Report," JPL Technical Document D-13388, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, January 1996.

***Steve Chien** is Technical Group Supervisor of the Artificial Intelligence Group of the Jet Propulsion Laboratory, California Institute of Technology where he leads efforts in research and development of automated planning and scheduling systems. He is also an adjunct assistant professor in the Department of Computer Science at the University of Southern California. He holds a B.S., M.S., and Ph.D. in Computer Science from the University of Illinois. His research interests are in the areas of: planning and scheduling, operations research, and machine learning.*

***Anita Govindjee** is a Member of Technical Staff in the Artificial Intelligence Group at the Jet Propulsion Laboratory, California Institute of Technology. She holds a M.S. in Computer Science from Stanford University and a B.S. in Computer Science from the University of Illinois. Her research interests are in artificial intelligence and cognitive science.*

***Tara Estlin** is a Ph.D. Candidate in Computer Science at the University of Texas at Austin. She holds a B.S. in Computer Science from Tulane University and a M.S. in Computer Science from the University of Texas at Austin. Her research interests include machine learning and planning and scheduling.*

***XueMei Wang** is a Member of Technical Staff at Rockwell Science Center in Palo Alto, California. She holds a Ph.D. in Computer Science from Carnegie-Mellon University and a B.S in Applied Mathematics from Tsinghua University. Her research interests are Artificial Intelligence including planning, machine learning, and knowledge acquisition.*

Randall Hill Jr. *is a Research Computer Scientist at the Information Sciences Institute, University of Southern California. He* **is also a** *Research Assistant* **Professor in** *the* **USC** *Department* **of Computer** *Science.* **Dr.** *Hill holds a* **B.S. from** *the United States* **Military** *Academy at* **West** *Point, and* **M. S.** *and Ph. D. degrees in Computer Science from the University of Southern California. His research interests lie in the areas of intelligent agents and computer-assisted learning environments.*